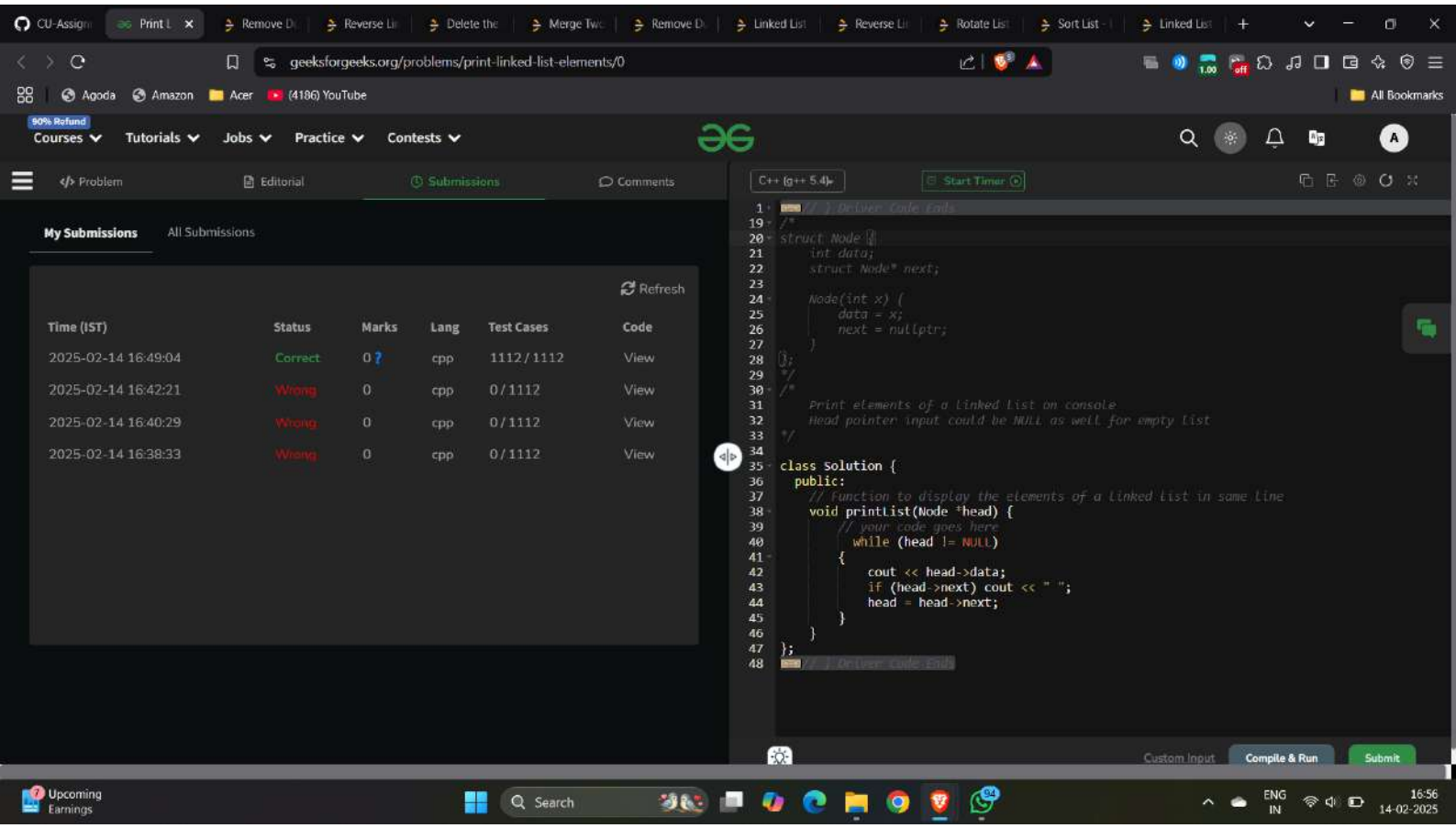
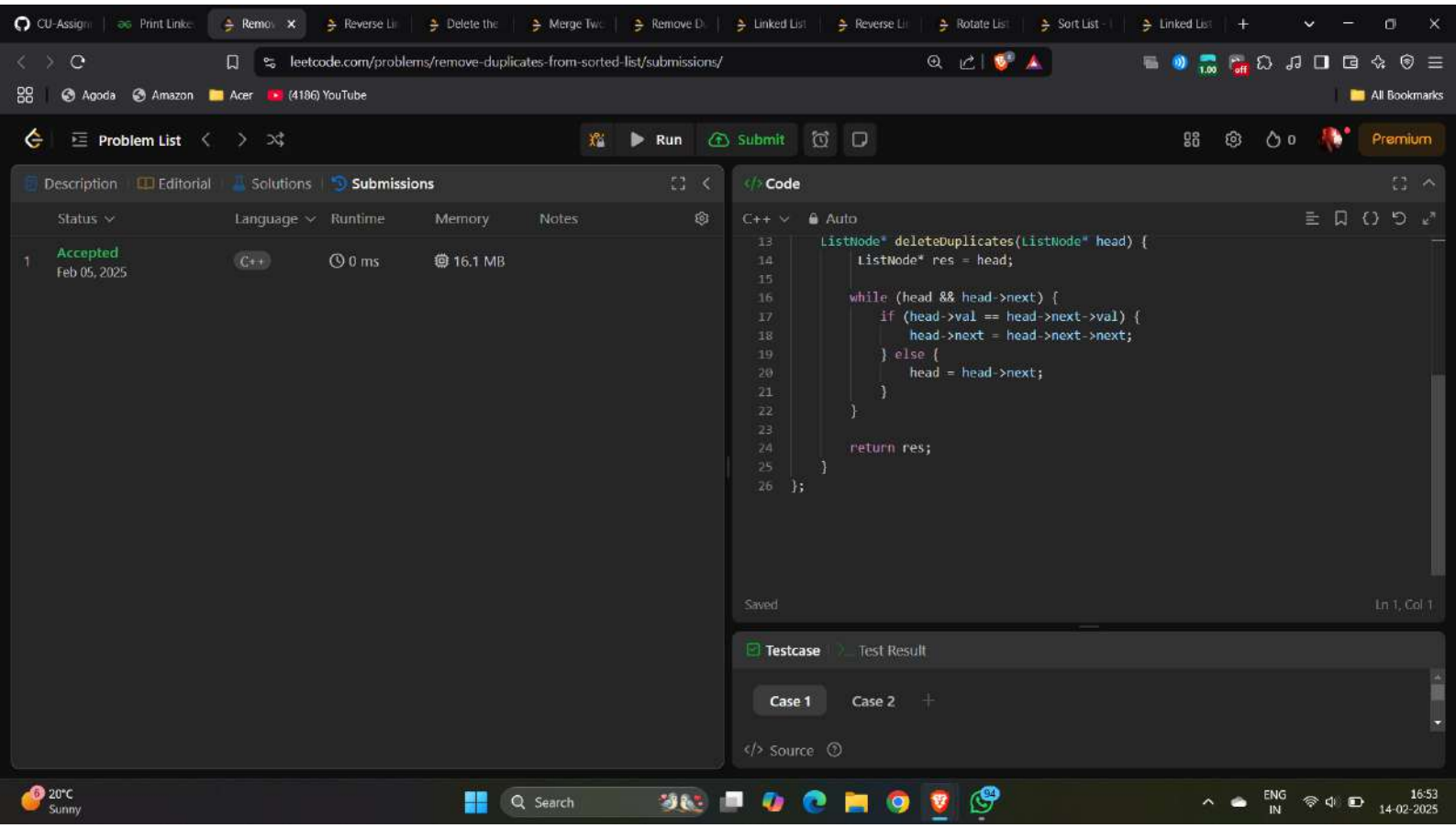
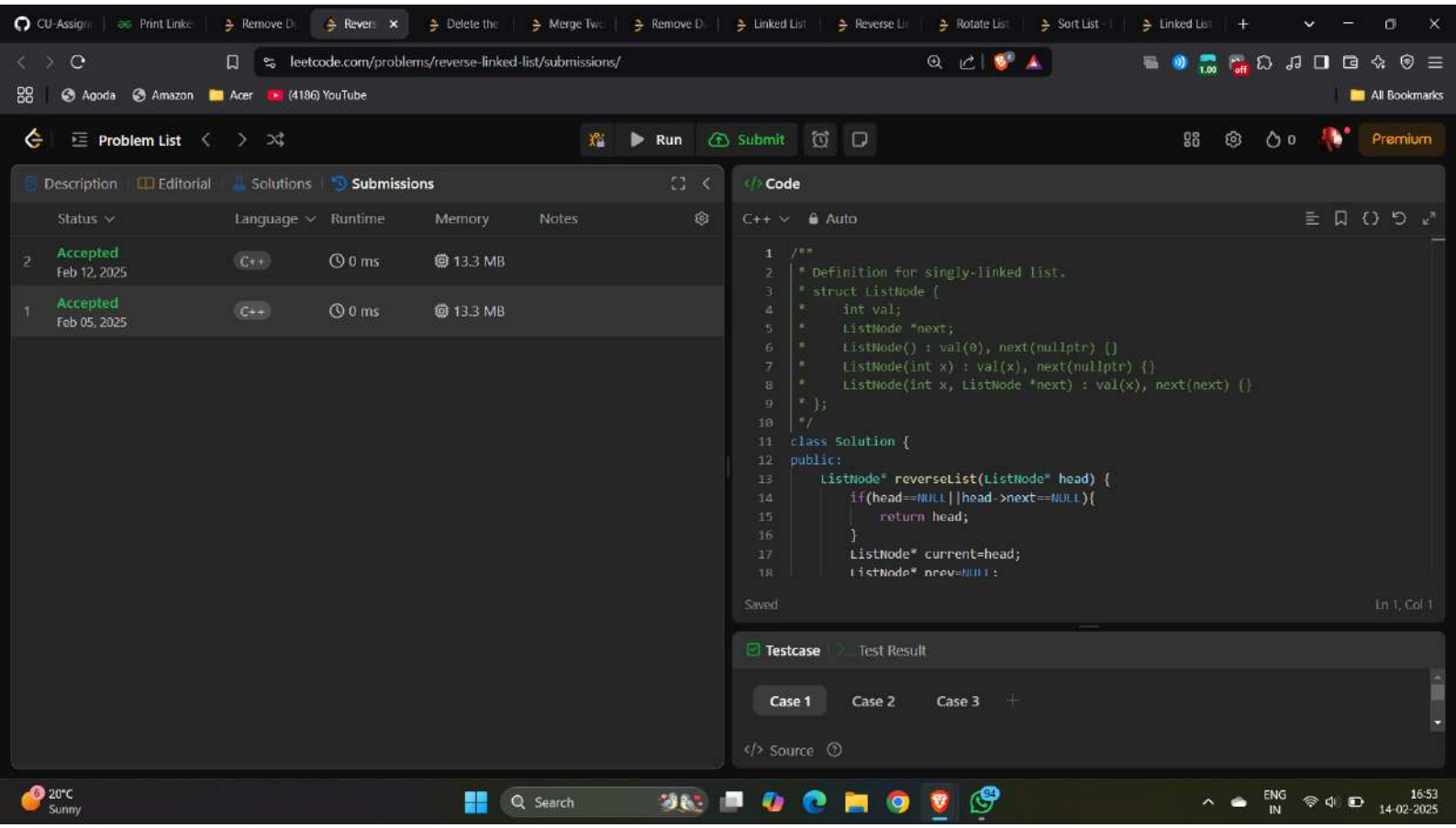
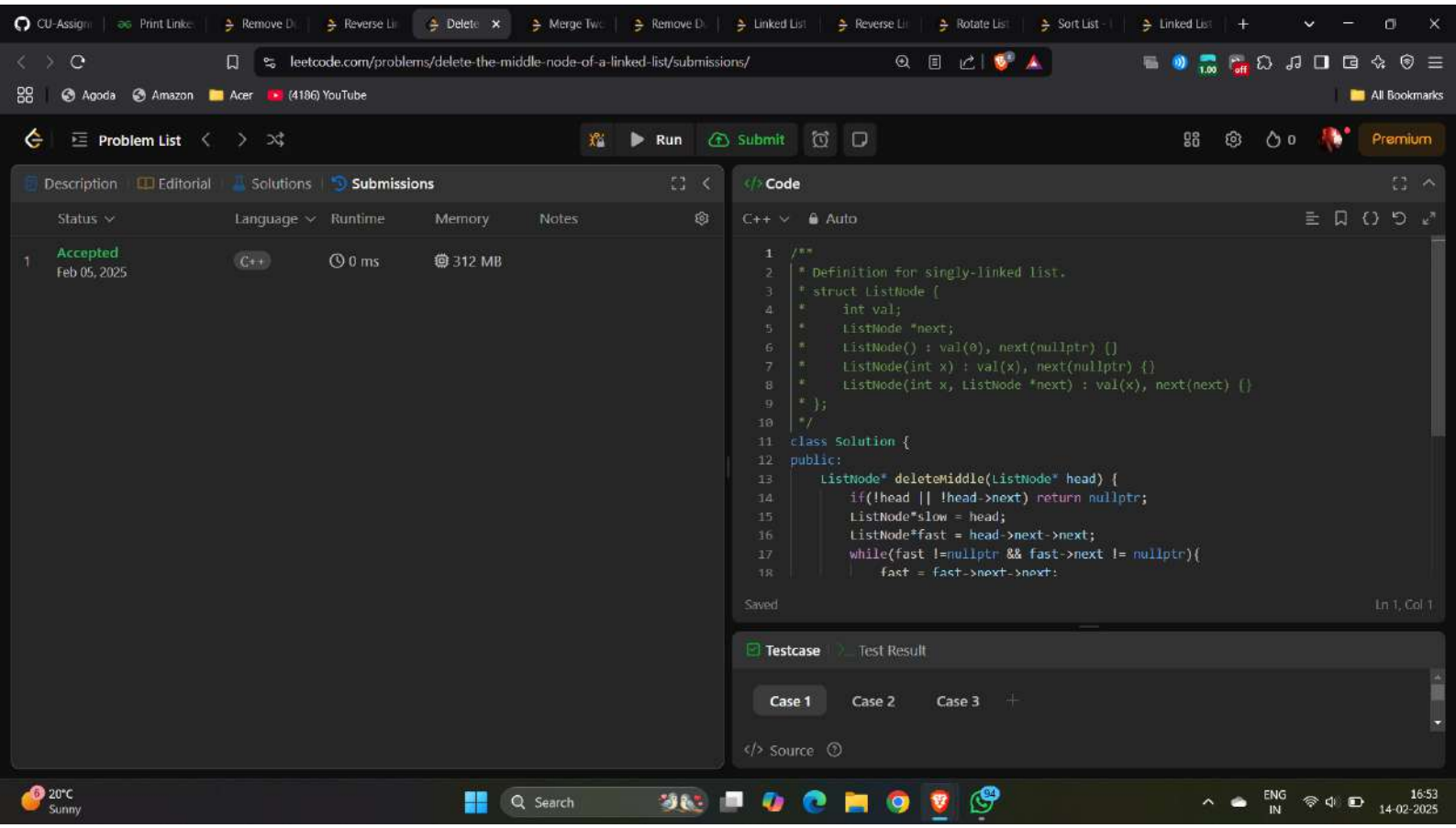
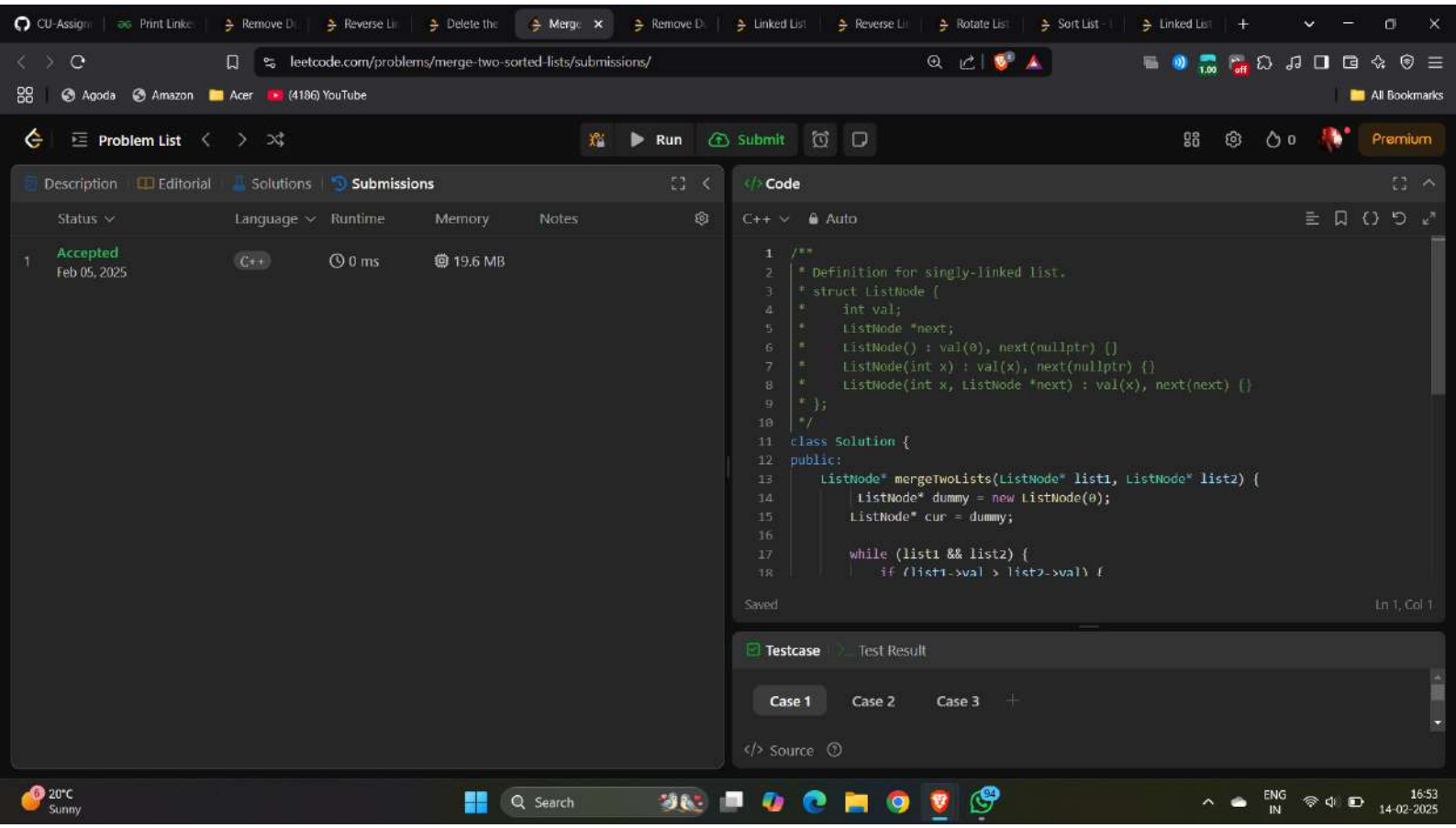
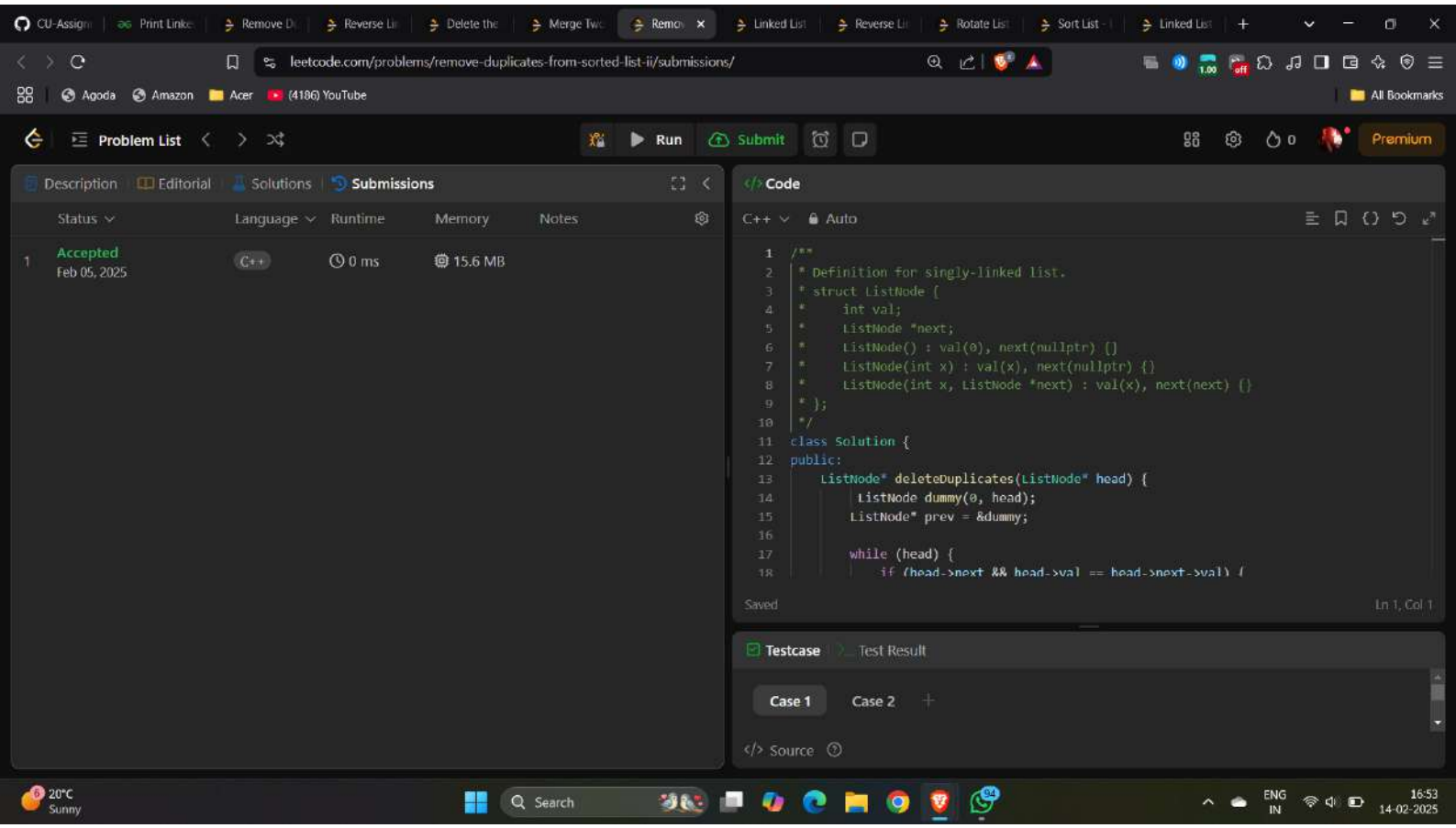

Print L
Remove D
Reverse Li
Delete the
Merge Two
Remove D
Linked List
Reverse Li
Rotate List
Sort List
Linked List
geeksforgeeks.org/problems/print-linked-list-elements/0
Agoda
Amazon
Acer
(4186) YouTube
All Bookmarks
90% Refund
Courses
Tutorials
Jobs
Practice
Contests
Problem
Editorial
Submissions
Comments
My Submissions
All Submissions
Refresh
Time (IST)
Status
Marks
Lang
Test Cases
Code
2025-02-14 16:49:04
Correct
0
cpp
1112 / 1112
View
2025-02-14 16:42:21
Wrong
0
cpp
0 / 1112
View
2025-02-14 16:40:29
Wrong
0
cpp
0 / 1112
View
2025-02-14 16:38:33
Wrong
0
cpp
0 / 1112
View
C++ (g++ 5.4)
Start Timer
// } Driver Code Ends
/*
struct Node {
int data;
struct Node* next;
Node(int x) {
data = x;
next = nullptr;
}
};
*/
/*
Print elements of a linked list on console
Head pointer input could be NULL as well for empty list
*/
class Solution {
public:
// Function to display the elements of a linked list in same line
void printList(Node *head) {
// your code goes here
while (head != NULL)
{
cout << head->data;
if (head->next) cout << " ";
head = head->next;
}
}
};
// } Driver Code Ends
Custom Input
Compile & Run
Submit
Upcoming Earnings
Search
ENG IN
16:56
14-02-2025

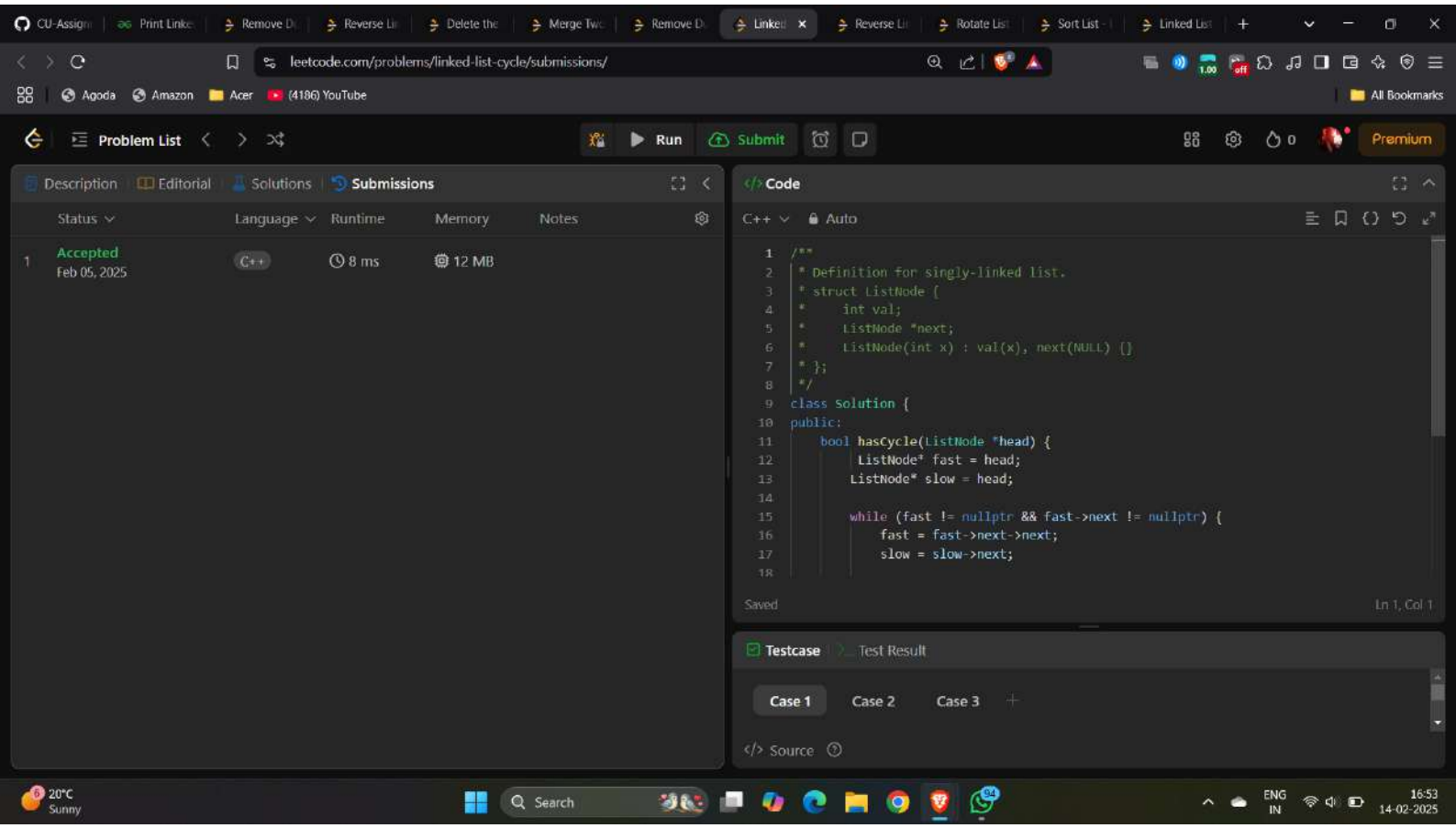

leetcode.com/problems/remove-duplicates-from-sorted-list/submissions/
Problem List
Run
Submit
Premium
Description
Editorial
Solutions
Submissions
Status
Language
Runtime
Memory
Notes
1
Accepted
Feb 05, 2025
C++
0 ms
16.1 MB
Code
C++
Auto
ListNode* deleteDuplicates(ListNode* head) {
ListNode* res = head;
while (head && head->next) {
if (head->val == head->next->val) {
head->next = head->next->next;
} else {
head = head->next;
}
}
return res;
}
};
Saved
Ln 1, Col 1
Testcase
Test Result
Case 1
Case 2
Source
20°C
Sunny
Search
ENG
IN
16:53
14-02-2025

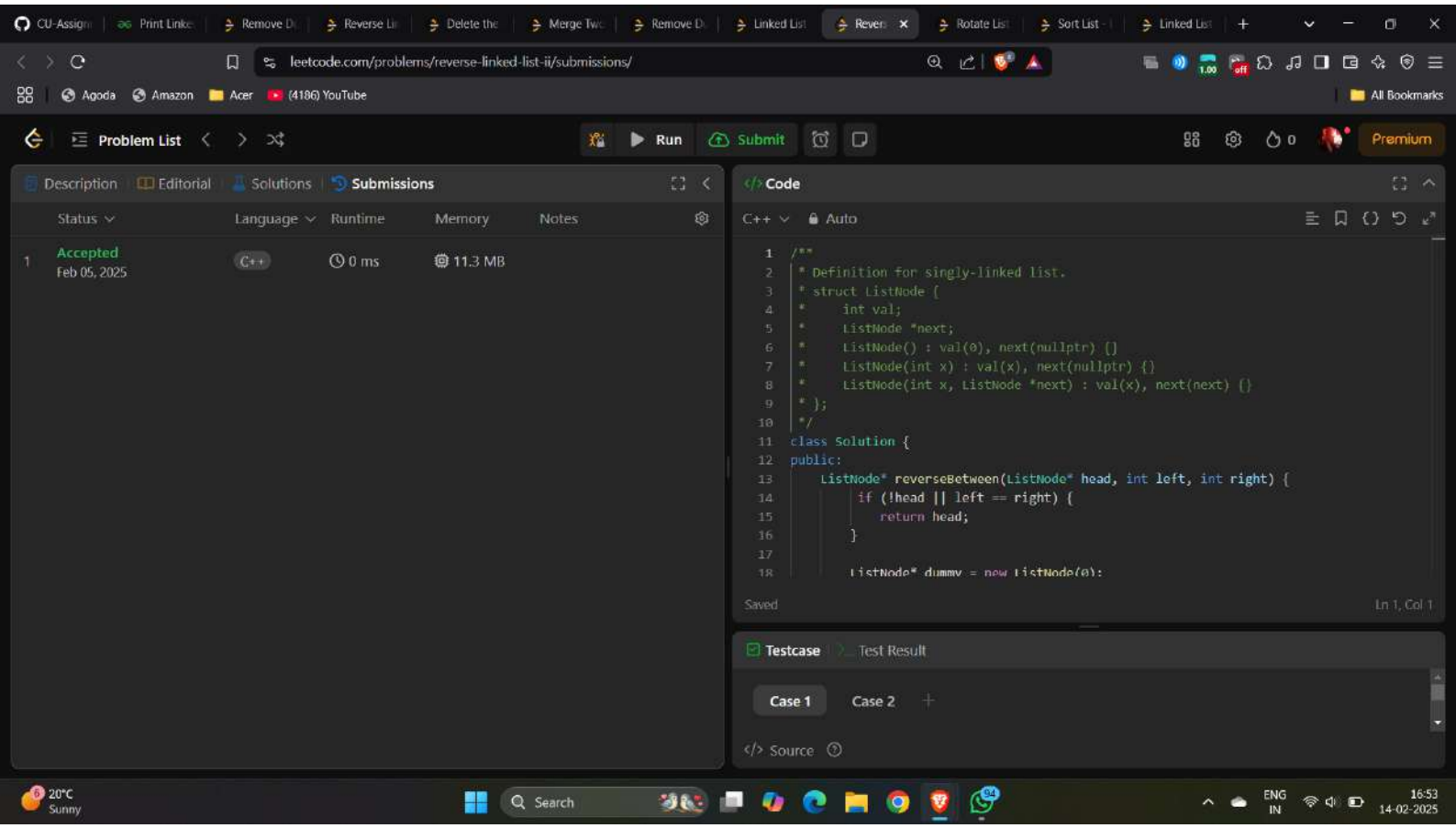

leetcode.com/problems/reverse-linked-list/submissions/
Problem List
Run
Submit
Premium
Description
Editorial
Solutions
Submissions
Status
Language
Runtime
Memory
Notes
2 Accepted Feb 12, 2025 C++ 0 ms 13.3 MB
1 Accepted Feb 05, 2025 C++ 0 ms 13.3 MB
Code
C++
Auto
/**
* Definition for singly-linked list.
* struct ListNode {
* int val;
* ListNode *next;
* ListNode() : val(0), next(nullptr) {}
* ListNode(int x) : val(x), next(nullptr) {}
* ListNode(int x, ListNode *next) : val(x), next(next) {}
* };
*/
class Solution {
public:
ListNode* reverseList(ListNode* head) {
if(head==NULL||head->next==NULL){
return head;
}
ListNode* current=head;
Saved
Ln 1, Col 1
Testcase
Test Result
Case 1
Case 2
Case 3
Source
20°C Sunny
Search
ENG IN
16:53 14-02-2025

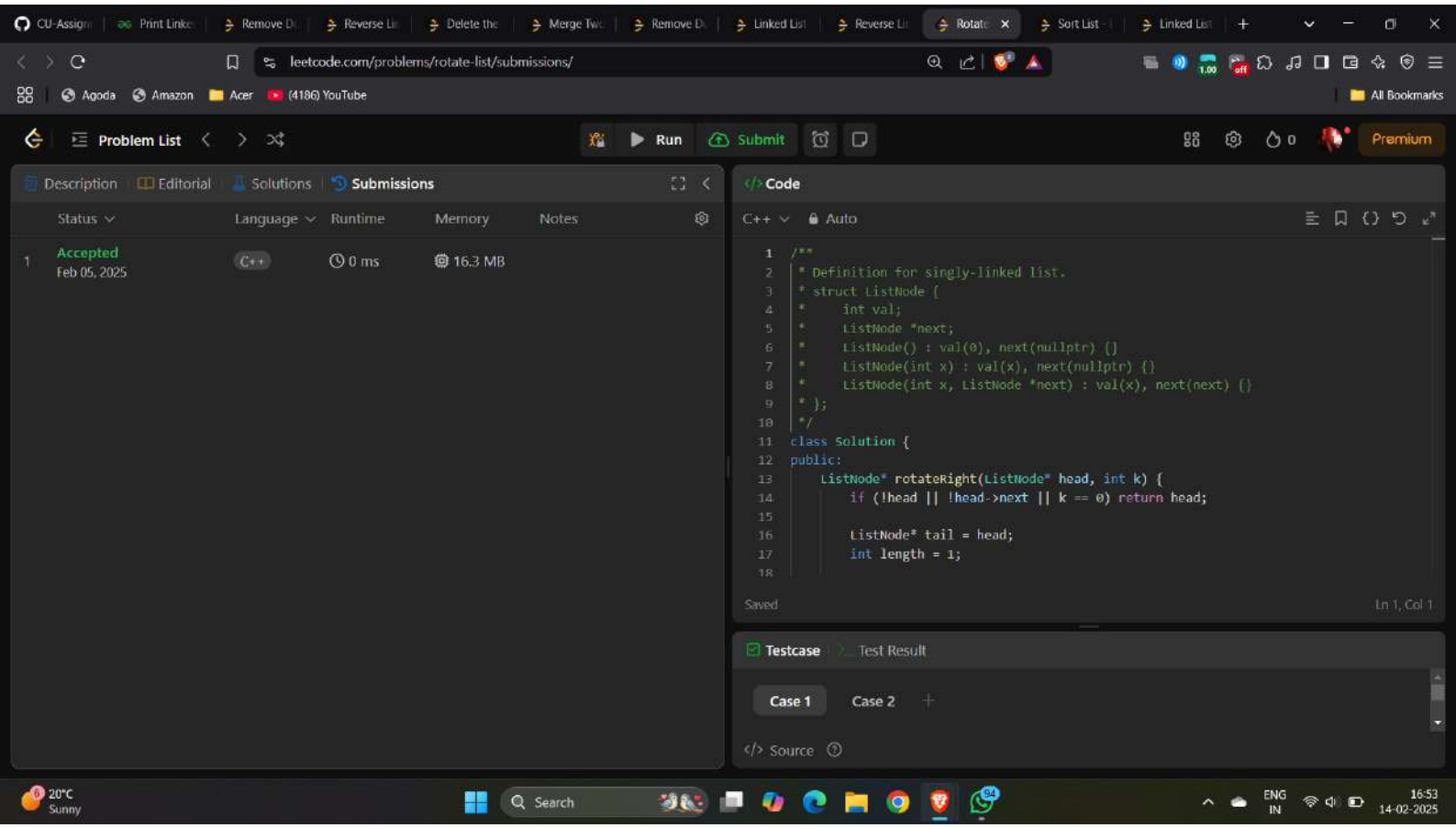

leetcode.com/problems/delete-the-middle-node-of-a-linked-list/submissions/
Problem List
Run
Submit
Premium
Description
Editorial
Solutions
Submissions
Status
Language
Runtime
Memory
Notes
1
Accepted
Feb 05, 2025
C++
0 ms
312 MB
Code
C++
Auto
/**
* Definition for singly-linked list.
* struct ListNode {
*   int val;
*   ListNode *next;
*   ListNode() : val(0), next(nullptr) {}
*   ListNode(int x) : val(x), next(nullptr) {}
*   ListNode(int x, ListNode *next) : val(x), next(next) {}
* };
*/
class Solution {
public:
    ListNode* deleteMiddle(ListNode* head) {
        if(!head || !head->next) return nullptr;
        ListNode*slow = head;
        ListNode*fast = head->next->next;
        while(fast !=nullptr && fast->next != nullptr){
Saved
Ln 1, Col 1
Testcase
Test Result
Case 1
Case 2
Case 3
Source
20°C
Sunny
Search
ENG
IN
16:53
14-02-2025

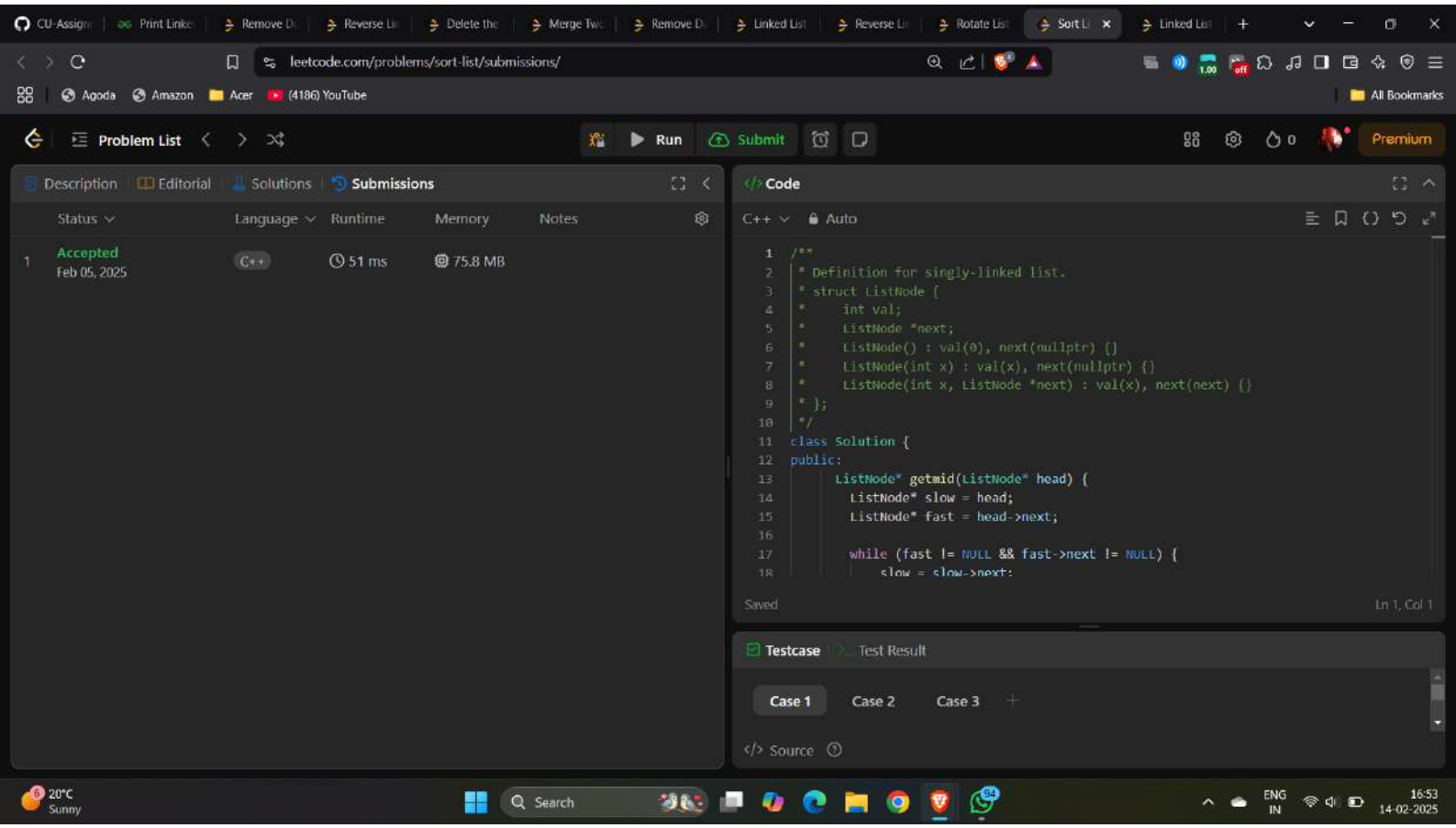

leetcode.com/problems/merge-two-sorted-lists/submissions/
Problem List
Run
Submit
Premium
Description
Editorial
Solutions
Submissions
Status
Language
Runtime
Memory
Notes
1
Accepted
Feb 05, 2025
C++
0 ms
19.6 MB
Code
C++
Auto
/**
* Definition for singly-linked list.
* struct ListNode {
*   int val;
*   ListNode *next;
*   ListNode() : val(0), next(nullptr) {}
*   ListNode(int x) : val(x), next(nullptr) {}
*   ListNode(int x, ListNode *next) : val(x), next(next) {}
* };
*/
class Solution {
public:
ListNode* mergeTwoLists(ListNode* list1, ListNode* list2) {
ListNode* dummy = new ListNode(0);
ListNode* cur = dummy;
while (list1 && list2) {
Saved
Ln 1, Col 1
Testcase
Test Result
Case 1
Case 2
Case 3
Source

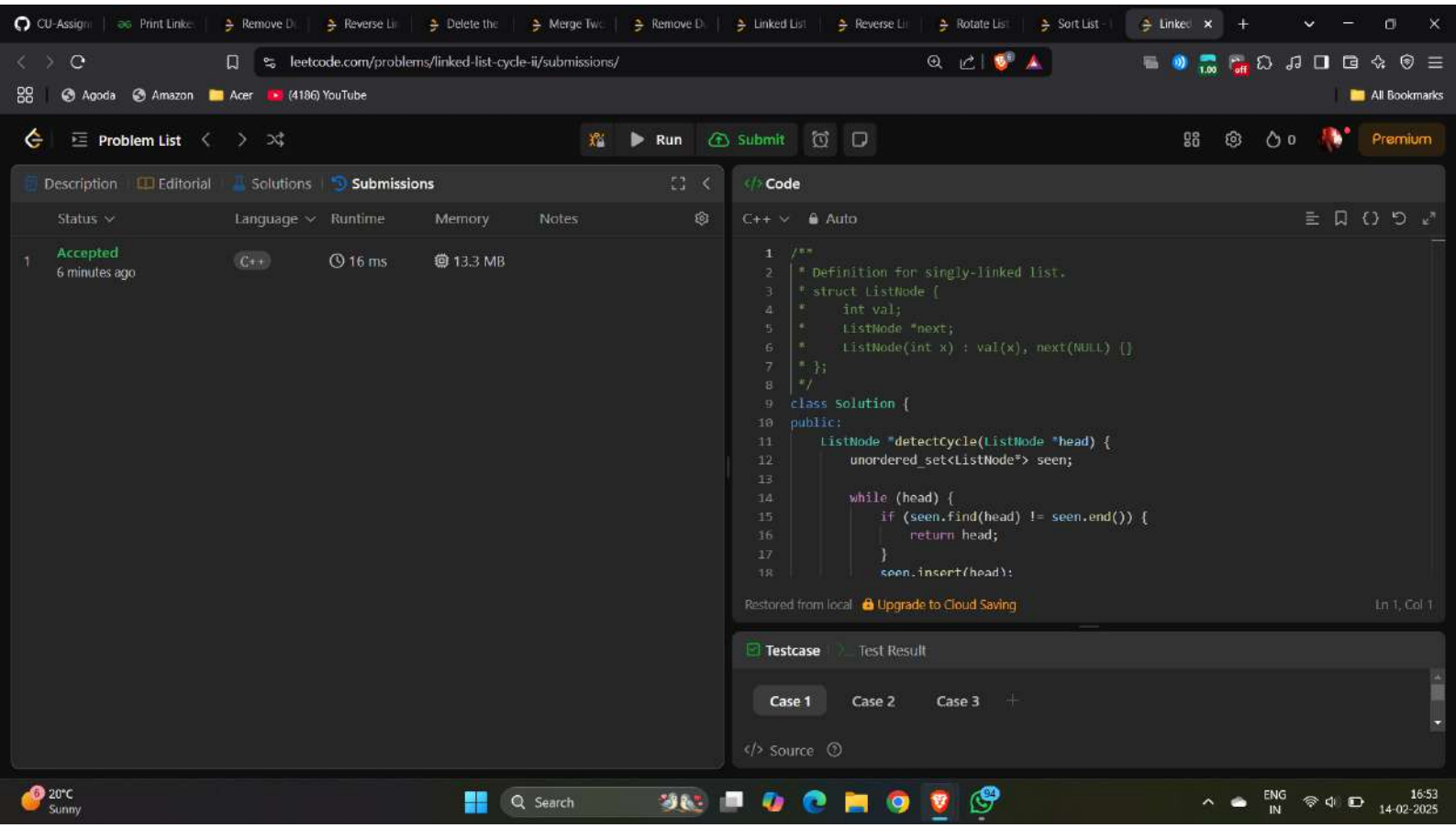

leetcode.com/problems/remove-duplicates-from-sorted-list-ii/submissions/
Problem List
Run
Submit
Premium
Description
Editorial
Solutions
Submissions
Status
Language
Runtime
Memory
Notes
1
Accepted
Feb 05, 2025
C++
0 ms
15.6 MB
Code
C++
Auto
/**
* Definition for singly-linked list.
* struct ListNode {
*   int val;
*   ListNode *next;
*   ListNode() : val(0), next(nullptr) {}
*   ListNode(int x) : val(x), next(nullptr) {}
*   ListNode(int x, ListNode *next) : val(x), next(next) {}
* };
*/
class Solution {
public:
ListNode* deleteDuplicates(ListNode* head) {
ListNode dummy(0, head);
ListNode* prev = &dummy;
while (head) {
if (head->next && head->val == head->next->val) {
Saved
Ln 1, Col 1
Testcase
Test Result
Case 1
Case 2
Source
20°C
Sunny
Search
ENG
IN
16:53
14-02-2025


leetcode.com/problems/linked-list-cycle/submissions/
Problem List
Run
Submit
Premium
Description
Editorial
Solutions
Submissions
Status
Language
Runtime
Memory
Notes
1
Accepted
Feb 05, 2025
C++
8 ms
12 MB
Code
C++
Auto
/**
* Definition for singly-linked list.
* struct ListNode {
* int val;
* ListNode *next;
* ListNode(int x) : val(x), next(NULL) {}
* };
*/
class Solution {
public:
bool hasCycle(ListNode *head) {
ListNode* fast = head;
ListNode* slow = head;
while (fast != nullptr && fast->next != nullptr) {
fast = fast->next->next;
slow = slow->next;
Saved
Ln 1, Col 1
Testcase
Test Result
Case 1
Case 2
Case 3
Source
20°C
Sunny
Search
ENG
IN
16:53
14-02-2025


leetcode.com/problems/reverse-linked-list-ii/submissions/
Problem List
Run
Submit
Premium
Description
Editorial
Solutions
Submissions
Status
Language
Runtime
Memory
Notes
Accepted
Feb 05, 2025
C++
0 ms
11.3 MB
Code
C++
Auto
/**
* Definition for singly-linked list.
* struct ListNode {
*   int val;
*   ListNode *next;
*   ListNode() : val(0), next(nullptr) {}
*   ListNode(int x) : val(x), next(nullptr) {}
*   ListNode(int x, ListNode *next) : val(x), next(next) {}
* };
*/
class Solution {
public:
ListNode* reverseBetween(ListNode* head, int left, int right) {
if (!head || left == right) {
return head;
}
ListNode* dummy = new ListNode(0);
Saved
Ln 1, Col 1
Testcase
Test Result
Case 1
Case 2
Source
20°C
Sunny
Search
ENG
IN
16:53
14-02-2025


leetcode.com/problems/rotate-list/submissions/
Problem List
Run
Submit
Premium
Description
Editorial
Solutions
Submissions
Status
Language
Runtime
Memory
Notes
1
Accepted
Feb 05, 2025
C++
0 ms
16.3 MB
Code
C++
Auto
/**
* Definition for singly-linked list.
* struct ListNode {
*   int val;
*   ListNode *next;
*   ListNode() : val(0), next(nullptr) {}
*   ListNode(int x) : val(x), next(nullptr) {}
*   ListNode(int x, ListNode *next) : val(x), next(next) {}
* };
*/
class Solution {
public:
    ListNode* rotateRight(ListNode* head, int k) {
        if (!head || !head->next || k == 0) return head;
        ListNode* tail = head;
        int length = 1;
Saved
Ln 1, Col 1
Testcase
Test Result
Case 1
Case 2
Source
20°C
Sunny
Search
ENG
IN
16:53
14-02-2025


leetcode.com/problems/sort-list/submissions/
Problem List
Run
Submit
Premium
Description
Editorial
Solutions
Submissions
Status
Language
Runtime
Memory
Notes
1
Accepted
Feb 05, 2025
C++
51 ms
75.8 MB
Code
C++
Auto
/**
* Definition for singly-linked list.
* struct ListNode {
*   int val;
*   ListNode *next;
*   ListNode() : val(0), next(nullptr) {}
*   ListNode(int x) : val(x), next(nullptr) {}
*   ListNode(int x, ListNode *next) : val(x), next(next) {}
* };
*/
class Solution {
public:
ListNode* getmid(ListNode* head) {
ListNode* slow = head;
ListNode* fast = head->next;
while (fast != NULL && fast->next != NULL) {
Saved
Ln 1, Col 1
Testcase
Test Result
Case 1
Case 2
Case 3
Source
20°C
Sunny
Search
ENG
IN
16:53
14-02-2025


leetcode.com/problems/linked-list-cycle-ii/submissions/
Problem List
Run
Submit
Premium
Description
Editorial
Solutions
Submissions
Status
Language
Runtime
Memory
Notes
1
Accepted
6 minutes ago
C++
16 ms
13.3 MB
Code
C++
Auto
/**
* Definition for singly-linked list.
* struct ListNode {
*   int val;
*   ListNode *next;
*   ListNode(int x) : val(x), next(NULL) {}
* };
*/
class Solution {
public:
ListNode *detectCycle(ListNode *head) {
unordered_set<ListNode*> seen;
while (head) {
if (seen.find(head) != seen.end()) {
return head;
}
Restored from local
Upgrade to Cloud Saving
Ln 1, Col 1
Testcase
Test Result
Case 1
Case 2
Case 3
Source